**Bazil Mupisiri**

**Exercise set #2**

**0)**

public class Main {  
  
 private int numOfComparisons = 0;  
  
 private int getNumOfComparisons(){  
 int numOfComparisons = this.numOfComparisons;  
 this.numOfComparisons = 0;  
 return numOfComparisons;  
 }  
  
 */\*\*  
 \* search for a string using Knuth Morris Pratt Algorithm  
 \** ***@param*** *needle, the string to be searched for  
 \** ***@param*** *haystack, the string we're searching  
 \** ***@return*** *boolean, true if the string is found and false otherwise  
 \*/* private boolean search(char[] needle, char[] haystack) {  
 int jump = 0;  
 if(needle.length <= haystack.length) {  
 for (int i = 0; i < haystack.length; i++) {  
 if (i + needle.length <= haystack.length) {  
  
 for (int x = 0; x < needle.length; x++) {  
 if (needle[x] == haystack[i + x] || needle[x] == '?') {  
 this.numOfComparisons++;  
 jump++;  
 if (jump == needle.length) {  
 return true;  
 }  
 }else {  
 break;  
 }  
 }//for  
 i += jump;  
 jump = 0;  
 }else {  
 break;  
 }  
 }//for  
 }  
 return false;  
 }//search  
  
 private static String resultToString(boolean result) {  
 return result?"'passed'":"'failed'";  
 }  
  
 private void printTest(String needle, String haystack){  
 boolean result = search(needle.toCharArray(), haystack.toCharArray());  
 System.*out*.println(  
 "Param: '" + needle + "' , Search Text: '" + haystack + "' , Result: "   
 + *resultToString*(result) + " with " + getNumOfComparisons() + " comparisons"  
 );  
 }  
  
 public static void main(String[] args) {  
 Main str = new Main();  
 System.*out*.println("Test Cases");  
 str.printTest("Hello", "Hello World");  
 str.printTest("He?lo", "Hello World");  
 str.printTest("Helly", "Hello World");  
 str.printTest("Hel lo", "Hello World");  
 str.printTest("", "Hello World");  
 str.printTest("Hello World", "Hello World");  
 str.printTest("World", "Hello World");  
 str.printTest("llo W", "Hello World");  
 str.printTest("Wo??d", "Hello World");  
 str.printTest("???", "Hello World");  
 }  
}

**2)**

**Test Cases**

Param: 'Hello' , Search Text: 'Hello World' , Result: 'passed' with 5 comparisons

Param: 'He?lo' , Search Text: 'Hello World' , Result: 'passed' with 5 comparisons

Param: 'Helly' , Search Text: 'Hello World' , Result: 'failed' with 4 comparisons

Param: 'Hel lo' , Search Text: 'Hello World' , Result: 'failed' with 3 comparisons

Param: '' , Search Text: 'Hello World' , Result: 'failed' with 0 comparisons

Param: 'Hello World' , Search Text: 'Hello World' , Result: 'passed' with 11 comparisons

Param: 'World' , Search Text: 'Hello World' , Result: 'passed' with 5 comparisons

Param: 'llo W' , Search Text: 'Hello World' , Result: 'passed' with 5 comparisons

Param: 'Wo??d' , Search Text: 'Hello World' , Result: 'passed' with 5 comparisons

Param: '???' , Search Text: 'Hello World' , Result: 'passed' with 3 comparisons

**3) Worst case = mn** and this occurs when the string to be searched is as large as the text to be searched.

**4)** **Worst case = mn** same as above because the ‘?’ is only standing in for any character so it won’t change anything